DRAFT − CUPS Software Programmers Manual

CUPS−SPM−1.0b1

Easy Software Products
Copyright 1997−1999, All Rights Reserved

Table of Contents

Preface **1**
System Overview 1
Document Overview 2

1 − Printing System Overview **3**
The Printing Problem 3
The Technology 4
Jobs 4
Classes 4
Filters 4
Printer Drivers 5
Networking 5

2 − The CUPS API **7**
The CUPS Library 7
Detecting the CUPS Library in Autoconf 7
Basic Services 7
Include Files 7
Getting the Available Printers and Classes 7
Printing Files 8
Setting Printer Options 8
Cancelling Jobs 8
HTTP Services 8
Include Files 8
Connecting to a Server 8
Setting Request Fields 8
Issuing a Request 8
Getting the Request Status 8
Sending Request Data 8
Reading Request Data 8
IPP Services 8
Include Files 9
Creating an IPP Request 9
Adding Attributes 9
Sending an IPP Request 9
Reading an IPP Response 9
Finding Attributes 9
Looping Through Attributes 9
IPP Standard Operations 9
IPP Extension Operations 9
CUPS Extension Operations 9
Language Services 9
Include Files 9
Getting the Default Language 10
Getting the Language Encoding 10
Getting a Language String 10
MIME Services 10

Table of Contents

Include Files....10
Loading a MIME Database....10
Finding a Specific MIME Type....10
Finding the MIME Type of a File....10
Filters....10
PPD Services....10
Include Files....10
Loading a PPD File....10
Options and Groups....11
Finding an Option....11
Finding a Page Size....11
Marking Options....11
Checking for Conflicts....11
Sending Options....11

3 − Writing Filters....13
Overview....13
Security Considerations....13
Temporary Files....13
Page Accounting....13
Command−Line Arguments....13
Copy Generation....14
Environment Variables....14
Writing a HTML Filter....14

4 − Writing Printer Drivers....15
Overview....15
Page Accounting....15
Color Management....15
Raster Functions....15
cupsRasterOpen()....15
cupsRasterReadHeader()....16
cupsRasterReadPixels()....16
cupsRasterClose()....16
Writing a HP−PCL Driver....16

5 − Writing Backends....17
Overview....17
Security Considerations....17
Temporary Files....17
Page Accounting....17
Retries....17
Command−Line Arguments....18
Copy Generation....18
Environment Variables....18
Writing a Serial Port Backend....18

Table of Contents

A − Constants **19**
CUPS Constants 19
HTTP Constants 19
IPP Constants 19
Language Constants 19
MIME Constants 20
PPD Constants 20
Raster Constants 20

B − Structures **21**

C − Functions **23**
cupsAddOption() 24
Usage 24
Arguments 24
Returns 24
Description 24
Example 24
See Also 25
cupsCancelJob() 26
Usage 26
Arguments 26
Returns 26
Description 26
Example 26
See Also 26
cupsDoFileRequest() 27
Usage 27
Arguments 27
Returns 27
Description 27
Example 27
See Also 28
cupsDoRequest() 29
Usage 29
Arguments 29
Returns 29
Description 29
Example 29
See Also 30
cupsFreeOptions() 31
Usage 31
Arguments 31
Description 31
Example 31
See Also 31
cupsGetClasses() 32

Table of Contents

Usage....32
Arguments....32
Returns....32
Description....32
Example....32
See Also....33
cupsGetDefault()....34
Usage....34
Returns....34
Description....34
Example....34
See Also....34
cupsGetOption()....35
Usage....35
Arguments....35
Returns....35
Description....35
See Also....35
cupsGetPassword()....36
Usage....36
Arguments....36
Returns....36
Description....36
Example....36
See Also....36
cupsGetPPD()....37
Usage....37
Arguments....37
Returns....37
Description....37
Example....37
cupsGetPrinters()....38
Usage....38
Arguments....38
Returns....38
Description....38
Example....38
See Also....39
cupsLangDefault()....40
Usage....40
Returns....40
Description....40
Example....40
See Also....40
cupsLangEncoding()....41
Usage....41
Arguments....41

Table of Contents

Returns 41
Description 41
Example 41
See Also 41
cupsLangFlush() 42
Usage 42
Description 42
Example 42
See Also 42
cupsLangFree() 43
Usage 43
Arguments 43
Description 43
Example 43
See Also 43
cupsLangGet() 44
Usage 44
Arguments 44
Returns 44
Description 44
Example 44
See Also 44
cupsLangString() 45
Usage 45
Arguments 45
Returns 45
Description 45
Example 45
See Also 45
cupsLastError() 46
Usage 46
Returns 46
Description 46
Example 46
See Also 46
cupsMarkOptions() 47
Usage 47
Arguments 47
Returns 47
Description 47
Example 47
See Also 47
cupsParseOptions() 48
Usage 48
Arguments 48
Returns 48
Description 48

Table of Contents

Example....48
See Also....48
cupsPrintFile()....49
Usage....49
Arguments....49
Returns....49
Description....49
Example....49
See Also....50
cupsRasterClose()....51
Usage....51
Arguments....51
Description....51
Example....51
See Also....51
cupsRasterOpen()....52
Usage....52
Arguments....52
Returns....52
Description....52
Example....52
See Also....52
cupsRasterReadHeader()....53
Usage....53
Arguments....53
Returns....53
Description....53
Example....53
See Also....54
cupsRasterReadPixels()....55
Usage....55
Arguments....55
Returns....55
Description....55
Example....55
See Also....56
cupsRasterWriteHeader()....57
Usage....57
Arguments....57
Returns....57
Description....57
Example....57
See Also....57
cupsRasterWritePixels()....58
Usage....58
Arguments....58
Returns....58

Table of Contents

Description....58
Example....58
See Also....59
cupsServer()....60
Usage....60
Returns....60
Description....60
Example....60
See Also....60
cupsTempFile()....61
Usage....61
Arguments....61
Returns....61
Description....61
Example....61
cupsUser()....62
Usage....62
Returns....62
Description....62
Example....62
See Also....62
httpBlocking()....63
Usage....63
Arguments....63
Returns....63
Description....63
Example....63
See Also....63
httpCheck()....64
Usage....64
Arguments....64
Returns....64
Description....64
Example....64
See Also....64
httpClearFields()....65
Usage....65
Arguments....65
Returns....65
Description....65
Example....65
See Also....65
httpClose()....66
Usage....66
Arguments....66
Returns....66
Description....66

Table of Contents

Example 66
See Also 66
httpConnect() 67
Usage 67
Arguments 67
Returns 67
Description 67
Example 67
See Also 67
httpDecode64() 68
Usage 68
Arguments 68
Returns 68
Description 68
Example 68
See Also 68
httpDelete() 69
Usage 69
Arguments 69
Returns 69
Description 69
Example 69
See Also 69
httpEncode64() 70
Usage 70
Arguments 70
Returns 70
Description 70
Example 70
See Also 70
httpError() 71
Usage 71
Arguments 71
Returns 71
Description 71
Example 71
See Also 71
httpFlush() 72
Usage 72
Arguments 72
Returns 72
Description 72
Example 72
See Also 72
httpGet() 73
Usage 73
Arguments 73

Table of Contents

Returns....73
Description....73
Example....73
See Also....73
httpGets()....74
Usage....74
Arguments....74
Returns....74
Description....74
Example....74
See Also....74
httpGetDateString()....75
Usage....75
Arguments....75
Returns....75
Description....75
Example....75
See Also....75
httpGetDateTime()....76
Usage....76
Arguments....76
Returns....76
Description....76
Example....76
See Also....76
httpGetField()....77
Usage....77
Arguments....77
Returns....77
Description....77
Example....77
See Also....77
httpGetLength()....78
Usage....78
Arguments....78
Returns....78
Description....78
Example....78
See Also....78
httpHead()....79
Usage....79
Arguments....79
Returns....79
Description....79
Example....79
See Also....79
httpInitialize()....80

Table of Contents

Usage....80
Arguments....80
Returns....80
Description....80
Example....80
See Also....80
httpOptions()....81
Usage....81
Arguments....81
Returns....81
Description....81
Example....81
See Also....81
httpPost()....82
Usage....82
Arguments....82
Returns....82
Description....82
Example....82
See Also....82
httpPrintf()....83
Usage....83
Arguments....83
Returns....83
Description....83
Example....83
See Also....83
httpPut()....84
Usage....84
Arguments....84
Returns....84
Description....84
Example....84
See Also....84
httpRead()....85
Usage....85
Arguments....85
Returns....85
Description....85
Example....85
See Also....85
httpReconnect()....86
Usage....86
Arguments....86
Returns....86
Description....86
Example....86

Table of Contents

See Also 86
httpSeparate() 87
Usage 87
Arguments 87
Returns 87
Description 87
Example 87
See Also 87
httpSetField() 88
Usage 88
Arguments 88
Returns 88
Description 88
Example 88
See Also 88
httpTrace() 89
Usage 89
Arguments 89
Returns 89
Description 89
Example 89
See Also 89
httpUpdate() 90
Usage 90
Arguments 90
Returns 90
Description 90
Example 90
See Also 90
httpWrite() 91
Usage 91
Arguments 91
Returns 91
Description 91
Example 91
See Also 91
ippAddBoolean() 92
Usage 92
Arguments 92
Returns 92
Description 92
Example 92
See Also 92
ippAddBooleans() 93
Usage 93
Arguments 93
Returns 93

Table of Contents

Description....93
Example....93
See Also....93
ippAddDate()....94
Usage....94
Arguments....94
Returns....94
Description....94
Example....94
See Also....94
ippAddInteger()....95
Usage....95
Arguments....95
Returns....95
Description....95
Example....95
See Also....95
ippAddIntegers()....96
Usage....96
Arguments....96
Returns....96
Description....96
Example....96
See Also....96
ippAddRange()....97
Usage....97
Arguments....97
Returns....97
Description....97
Example....97
See Also....97
ippAddRanges()....98
Usage....98
Arguments....98
Returns....98
Description....98
Example....98
See Also....98
ippAddResolution()....99
Usage....99
Arguments....99
Returns....99
Description....99
Example....99
See Also....99
ippAddResolutions()....100
Usage....100

Table of Contents

Arguments....100
Returns....100
Description....100
Example....100
See Also....100
ippAddSeparator()....101
Usage....101
Arguments....101
Returns....101
Description....101
Example....101
See Also....101
ippAddString()....102
Usage....102
Arguments....102
Returns....102
Description....102
Example....102
See Also....102
ippAddStrings()....103
Usage....103
Arguments....103
Returns....103
Description....103
Example....103
See Also....103
ippDateToTime()....104
Usage....104
Arguments....104
Returns....104
Description....104
Example....104
See Also....104
ippDelete()....105
Usage....105
Arguments....105
Returns....105
Description....105
Example....105
See Also....105
ippFindAttribute()....106
Usage....106
Arguments....106
Returns....106
Description....106
Example....106
See Also....106

Table of Contents

ippLength() 107
Usage 107
Arguments 107
Returns 107
Description 107
Example 107
See Also 107
ippNew() 108
Usage 108
Arguments 108
Returns 108
Description 108
Example 108
See Also 108
ippPort() 109
Usage 109
Arguments 109
Returns 109
Description 109
Example 109
See Also 109
ippRead() 110
Usage 110
Arguments 110
Returns 110
Description 110
Example 110
See Also 110
ippTimeToDate() 111
Usage 111
Arguments 111
Returns 111
Description 111
Example 111
See Also 111
ippWrite() 112
Usage 112
Arguments 112
Returns 112
Description 112
Example 112
See Also 112
mimeAddFilter() 113
Usage 113
Arguments 113
Returns 113
Description 113

Table of Contents

Example....113
See Also....113
mimeAddType()....114
Usage....114
Arguments....114
Returns....114
Description....114
Example....114
See Also....114
mimeAddTypeRule()....115
Usage....115
Arguments....115
Returns....115
Description....115
Example....115
See Also....115
mimeDelete()....116
Usage....116
Arguments....116
Returns....116
Description....116
Example....116
See Also....116
mimeFileType()....117
Usage....117
Arguments....117
Returns....117
Description....117
Example....117
See Also....117
mimeFilter()....118
Usage....118
Arguments....118
Returns....118
Description....118
Example....118
See Also....118
mimeLoad()....119
Usage....119
Arguments....119
Returns....119
Description....119
Example....119
See Also....119
mimeMerge()....120
Usage....120
Arguments....120

Table of Contents

Returns....120
Description....120
Example....120
See Also....120
mimeNew()....121
Usage....121
Arguments....121
Returns....121
Description....121
Example....121
See Also....121
mimeType()....122
Usage....122
Arguments....122
Returns....122
Description....122
Example....122
See Also....122
ppdClose()....123
Usage....123
Arguments....123
Returns....123
Description....123
Example....123
See Also....123
ppdConflicts()....124
Usage....124
Arguments....124
Returns....124
Description....124
Example....124
See Also....124
pddEmitFd()....125
Usage....125
Arguments....125
Returns....125
Description....125
Example....125
See Also....125
ppdEmit()....126
Usage....126
Arguments....126
Returns....126
Description....126
Example....126
See Also....126
ppdFindChoice()....127

Table of Contents

Usage....127
Arguments....127
Returns....127
Description....127
Example....127
See Also....127
ppdFindMarkedChoice()....128
Usage....128
Arguments....128
Returns....128
Description....128
Example....128
See Also....128
ppdFindOption()....129
Usage....129
Arguments....129
Returns....129
Description....129
Example....129
See Also....129
ppdIsMarked()....130
Usage....130
Arguments....130
Returns....130
Description....130
Example....130
See Also....130
ppdMarkDefaults()....131
Usage....131
Arguments....131
Returns....131
Description....131
Example....131
See Also....131
ppdMarkOption()....132
Usage....132
Arguments....132
Returns....132
Description....132
Example....132
See Also....132
ppdOpenFd()....133
Usage....133
Arguments....133
Returns....133
Description....133
Example....133

Table of Contents

See Also 133
ppdOpenFile() 134
Usage 134
Arguments 134
Returns 134
Description 134
Example 134
See Also 134
ppdOpen() 135
Usage 135
Arguments 135
Returns 135
Description 135
Example 135
See Also 135
ppdPageLength() 136
Usage 136
Arguments 136
Returns 136
Description 136
Example 136
See Also 136
ppdPageSize() 137
Usage 137
Arguments 137
Returns 137
Description 137
Example 137
See Also 137
ppdPageWidth() 138
Usage 138
Arguments 138
Returns 138
Description 138
Example 138
See Also 138

Preface

This software programmers manual provides software programming information for the Common UNIX Printing System ("CUPS") Version 1.0.

System Overview

The Common UNIX Printing System provides a portable printing layer for UNIX® operating systems. It has been developed by Easy Software Products to promote a standard printing solution for all UNIX vendors and users. CUPS provides the System V and Berkeley command−line interfaces.

CUPS uses the Internet Printing Protocol (IETF−IPP) as the basis for managing print jobs and queues. The Line Printer Daemon (LPD, RFC1179), Server Message Block (SMB), and AppSocket protocols are also supported with reduced functionality.

CUPS adds network printer browsing and PostScript Printer Description ("PPD")−based printing options to support real world applications under UNIX.

CUPS also includes a customized version of GNU GhostScript (currently based off GNU GhostScript 4.03) and an image file RIP that can be used to support non−PostScript printers.

Document Overview

This software administrators manual is organized into the following sections:

- 1 − Printing System Overview
- 2 − The CUPS API
- 3 − Writing Filters
- 4 − Writing Printer Drivers
- 5 − Writing Backends
- A − Constants
- B − Structures
- C − Functions

1 – Printing System Overview

This chapter provides an overview of how the Common UNIX Printing System works.

The Printing Problem

For years *the printing problem* has plagued UNIX®. Unlike Microsoft® Windows® or MacOS, UNIX has no standard interface or system in place for supporting printers. Among the solutions previously available, the Berkeley and System V printing systems are the most prevalent.

These printing systems support line printers (text only) or PostScript printers (text and graphics), and with some coaxing they can be made to support a full range of printers and file formats. However, because each varient of the UNIX operating system uses a different printing system than the next, developing printer drivers for a wide range of printers is extremely difficult. That combined with the limited volume of customers for each UNIX varient has forced most printer vendors to give up supporting UNIX entirely.

The Common UNIX Printing System, or CUPS, is designed to eliminate *the printing problem*. One common printing system can be used by all UNIX varients to support the printing needs of users. Printer vendors can use its modular filter interface to develop a single driver program that supports a wide range of file formats with little or no effort. Since CUPS provides both the System V and Berkeley printing commands, users (and applications) can reap the benefits of this new technology with no changes.

The Technology

CUPS is based upon an emerging Internet standard called the Internet Printing Protocol, or IPP. IPP has been embraced by dozens of printer and printer server manufacturers, and will be supported by the next Microsoft Windows operating system.

IPP defines a standard protocol for printing as well as managing print jobs and printer options like media size, resolution, and so forth. Like all IP−based protocols, IPP can be used locally or over the Internet to printers hundreds or thousands of miles away. Unlike other protocols, however, IPP also supports access control, authentication, and encryption, making it a much more secure printing solution than older ones.

IPP is layered on top of the Hyper−Text Transport Protocol, or HTTP, which is the basis of web servers on the Internet. This allows the user to view documentation and status information on a printer or server using their web browser.

CUPS provides a complete IPP/1.0−based printing system that provides Basic authentication and domain or IP−based access control. Digest authentication and TLS encryption will be available in future versions of CUPS.

Jobs

Each file that is submitted for printing is called a *job*. Jobs are identified by a unique number starting at 1 and are assigned to a particular destination (usually a printer). Jobs can also have options associated with them such as media size, number of copies, and priority.

Classes

CUPS supports collections of printers known as *classes*. Jobs sent to a class are forwarded to the first available printer in the class.

Filters

Filters allow a user or application to print many types of files without extra effort. Print jobs sent to a CUPS server are filtered before sending them to a printer. Some filters convert job files to different formats that the printer can understand. Others perform page selection and ordering tasks. *Backend* filters perform the most important task of all − they send the filtered print data to the printer.

CUPS provides filters for printing many types of image files, HP−GL/2 files, PDF files, and text files. CUPS also supplies PostScript and image file Raster Image Processors, or RIPs, that convert PostScript or image files into bitmaps that can be sent to a raster printer.

CUPS provides backends for printing over parallel and serial ports, and over the network via the JetDirect (AppSocket), Server Message Block, and Line Printer Daemon protocols.

Printer Drivers

Printer drivers in CUPS consist of one of more filters specific to a printer. CUPS includes a sample printer driver for Hewlett−Packard LaserJet and DeskJet printers. While this driver does not generate optimal output for different printer models, it does demonstrate how you can write your own printer drivers and incorporate them into CUPS.

Networking

Printers and classes on the local system are automatically shared with other systems on the network. This allows you to setup one system to print to a printer and use this system as a printer server or spool host for all of the others. If there is only one occurrence of a printer on a network, then that printer can be accessed using its name alone. If more than one printer exists with the same name, users must select the printer by specifying which server to use (e.g. "printer@host1" or "printer@host2".)

CUPS also provides *implicit classes*, which are collections of printers and/or classes with the same name. This allows you to setup multiple servers pointing to the same physical network printer, for example, so that you aren't relying on a single system for printing. Because this also works with printer classes, you can setup multiple servers and printers and never worry about a "single point of failure" unless all of the printers and servers goes down!

2 − The CUPS API

This chapter describes the CUPS Application Programmers Interface ("API").

The CUPS Library

Detecting the CUPS Library in Autoconf

Basic Services

Include Files

Getting the Available Printers and Classes

Printing Files

Setting Printer Options

Cancelling Jobs

HTTP Services

Include Files

Connecting to a Server

Setting Request Fields

Issuing a Request

Getting the Request Status

Sending Request Data

Reading Request Data

IPP Services

Include Files

Creating an IPP Request

Adding Attributes

Sending an IPP Request

Reading an IPP Response

Finding Attributes

Looping Through Attributes

IPP Standard Operations

IPP Extension Operations

CUPS Extension Operations

Language Services

Include Files

Getting the Default Language

Getting the Language Encoding

Getting a Language String

MIME Services

Include Files

Loading a MIME Database

Finding a Specific MIME Type

Finding the MIME Type of a File

Filters

PPD Services

Include Files

Loading a PPD File

Options and Groups

Finding an Option

Finding a Page Size

Marking Options

Checking for Conflicts

Sending Options

3 − Writing Filters

This chapter describes how to write a file filter for CUPS.

Overview

Security Considerations

Users and Groups

Temporary Files

Page Accounting

Command−Line Arguments

Copy Generation

Environment Variables

Writing a HTML Filter

4 – Writing Printer Drivers

This chapter discusses how to write a printer driver, which is a special filter program that converts CUPS raster data into the appropriate commands and data required for a printer.

Overview

Page Accounting

Color Management

Raster Functions

cupsRasterOpen()

cupsRasterReadHeader()

cupsRasterReadPixels()

cupsRasterClose()

Writing a HP−PCL Driver

5 − Writing Backends

This chapter describes how to write a backend for CUPS. Backends communicate directly with printers and allow printer drivers and filters to send data using any type of connection transparently.

Overview

Security Considerations

Users and Groups

Temporary Files

Page Accounting

Retries

Command−Line Arguments

Copy Generation

Environment Variables

Writing a Serial Port Backend

A − Constants

This appendix lists all of the constants that are defined by the CUPS API.

CUPS Constants

HTTP Constants

IPP Constants

Language Constants

MIME Constants

PPD Constants

Raster Constants

B − Structures

This appendix describes all of the structures that are defined by the CUPS API.

C – Functions

This appendix provides a reference for all of the CUPS API functions.

cupsAddOption()

Usage

```
int
cupsAddOption(const char *name,
              const char *value,
              int num_options,
              cups_option_t **options);
```

Arguments

Argument	Description
name	The name of the option.
value	The value of the option.
num_options	Number of options currently in the array.
options	Pointer to the options array.

Returns

The new number of options.

Description

`cupsAddOption()` adds an option to the specified array.

Example

```
#include <cups.h>

...

/* Declare the options array */
int           num_options;
cups_option_t *options;

/* Initialize the options array */
num_options = 0;
options     = (cups_option_t *)0;

/* Add options using cupsAddOption() */
num_options = cupsAddOption("media", "letter", num_options, &options);
num_options = cupsAddOption("resolution", "300dpi", num_options, &options);
```

See Also

cupsFreeOptions(), cupsGetOption(), cupsParseOptions()

cupsCancelJob()

Usage

```
int
cupsCancelJob(const char *dest,
              int job);
```

Arguments

Argument	Description
dest	Printer or class name
job	Job ID

Returns

1 on success, 0 on failure. On failure the error can be found by calling `cupsLastError()`.

Description

`cupsCancelJob()` cancels the specifies job.

Example

```
#include <cups.h>

cupsCancelJob("LaserJet", 1);
```

See Also

`cupsLastError()`, `cupsPrintFile()`

cupsDoFileRequest()

Usage

```
ipp_t *
cupsDoFileRequest(http_t *http,
                  ipp_t *request,
                  const char *resource,
                  const char *filename);
```

Arguments

Argument	Description
http	HTTP connection to server.
request	IPP request data.
resource	HTTP resource name for POST.
filename	File to send with POST request (NULL pointer if none.)

Returns

IPP response data or NULL if the request fails. On failure the error can be found by calling cupsLastError().

Description

cupsDoFileRequest() does a HTTP POST request and provides the IPP request and optionally the contents of a file to the IPP server. It also handles resubmitting the request and performing password authentication as needed.

Example

```
#include <cups.h>

http_t      *http;
cups_lang_t *language;
ipp_t       *request;
ipp_t       *response;

...

/* Get the default language */
language = cupsLangDefault();

/* Create a new IPP request */
request  = ippNew();
```

```
request->request.op.operation_id = IPP_PRINT_FILE;
request->request.op.request_id   = 1;

/* Add required attributes */
ippAddString(request, IPP_TAG_OPERATION, IPP_TAG_CHARSET,
             "attributes-charset", NULL, cupsLangEncoding(language));

ippAddString(request, IPP_TAG_OPERATION, IPP_TAG_LANGUAGE,
             "attributes-natural-language", NULL,
             language != NULL ? language->language : "C");

ippAddString(request, IPP_TAG_OPERATION, IPP_TAG_URI, "printer-uri",
             NULL, "ipp://hostname/resource");

ippAddString(request, IPP_TAG_OPERATION, IPP_TAG_NAME, "requesting-user-name",
             NULL, cupsUser());

/* Do the request... */
response = cupsDoFileRequest(http, request, "/resource", "filename.txt");
```

See Also

cupsLangDefault(), cupsLangEncoding(), cupsUser(), httpConnect(), ippAddString(), ippNew()

cupsDoRequest()

Usage

```
ipp_t *
cupsDoRequest(http_t *http,
              ipp_t *request,
              const char *resource);
```

Arguments

Argument	Description
http	HTTP connection to server.
request	IPP request data.
resource	HTTP resource name for POST.

Returns

IPP response data or `NULL` if the request fails. On failure the error can be found by calling `cupsLastError()`.

Description

`cupsDoRequest()` does a HTTP POST request and provides the IPP request to the IPP server. It also handles resubmitting the request and performing password authentication as needed.

Example

```
#include <cups.h>

http_t      *http;
cups_lang_t *language;
ipp_t       *request;
ipp_t       *response;

...

/* Get the default language */
language = cupsLangDefault();

/* Create a new IPP request */
request  = ippNew();

request->request.op.operation_id = IPP_GET_PRINTER_ATTRIBUTES;
request->request.op.request_id   = 1;

/* Add required attributes */
```

```
ippAddString(request, IPP_TAG_OPERATION, IPP_TAG_CHARSET,
             "attributes-charset", NULL, cupsLangEncoding(language));

ippAddString(request, IPP_TAG_OPERATION, IPP_TAG_LANGUAGE,
             "attributes-natural-language", NULL,
             language != NULL ? language->language : "C");

ippAddString(request, IPP_TAG_OPERATION, IPP_TAG_URI, "printer-uri",
             NULL, "ipp://hostname/resource");

/* Do the request... */
response = cupsDoRequest(http, request, "/resource");
```

See Also

cupsLangDefault(), cupsLangEncoding(), cupsUser(), httpConnect(), ippAddString(), ippNew()

cupsFreeOptions()

Usage

```
void
cupsFreeOptions(int num_options,
                cups_option_t *options);
```

Arguments

Argument	Description
num_options	Number of options in array.
options	Pointer to options array.

Description

`cupsFreeOptions()` frees all memory associated with the option array specified.

Example

```
#include <cups/cups.h>

int           num_options;
cups_option_t *options;

...

cupsFreeOptions(num_options, options);
```

See Also

cupsAddOption(), cupsGetOption(), cupsMarkOptions(), cupsParseOptions()

cupsGetClasses()

Usage

```
int
cupsGetClasses(char ***classes);
```

Arguments

Argument	Description
classes	Pointer to character pointer array.

Returns

The number of printer classes available.

Description

`cupsGetClasses()` gets a list of the available printer classes. The returned array should be freed using the `free()` when it is no longer needed.

Example

```
#include <cups/cups.h>

int  i;
int  num_classes;
char **classes;

...

num_classes = cupsGetClasses(;

...

if (num_classes > 0)
{
  for (i = 0; i num_classes; i ++)
    free(classes[i]);

  free(classes);
}
```

See Also

cupsGetDefault(), cupsGetPrinters()

cupsGetDefault()

Usage

```
const char *
cupsGetDefault(void);
```

Returns

A pointer to the default destination.

Description

`cupsGetDefault()` gets the default destination printer or class. The default destination is stored in a static string and will be overwritten (usually with the same value) after each call.

Example

```
#include <cups/cups.h>

printf("The default destination is %s\n", cupsGetDefault());
```

See Also

cupsGetClasses(), cupsGetPrinters()

cupsGetOption()

Usage

```
const char *
cupsGetOption(const char *name,
              int num_options,
              cups_option_t *options);
```

Arguments

Argument	Description
name	The name of the option.
num_options	The number of options in the array.
options	The options array.

Returns

A pointer to the option values or `NULL` if the option is not defined.

Description

`cupsGetOption()` returns the first occurrence of the named option. If the option is not included in the options array then a `NULL` pointer is returned.

```
#include <cups/cups.h>

int           num_options;
cups_option_t *options;
const char    *media;

...

media = cupsGetOption("media", num_options, options);
```

See Also

cupsAddOption(), cupsFreeOptions(), cupsMarkOptions(), cupsParseOptions()

cupsGetPassword()

Usage

```
const char *
cupsGetPassword(const char *prompt);
```

Arguments

Argument	Description
prompt	The prompt to display to the user.

Returns

A pointer to the password that was entered or `NULL` if no password was entered.

Description

`cupsGetPassword()` displays the prompt string and asks the user for a password. The password text is not echoed to the user.

Example

```
#include <cups/cups.h>

char *password;

...

password = cupsGetPassword("Please enter a password:");
```

See Also

cupsServer(), cupsUser()

cupsGetPPD()

Usage

```
const char *
cupsGetPPD(const char *printer);
```

Arguments

Argument	Description
printer	The name of the printer.

Returns

The name of a temporary file containing the PPD file or `NULL` if the printer cannot be located or does not have a PPD file.

Description

`cupsGetPPD()` gets a copy of the PPD file for the named printer. The printer name can be of the form "printer" or "printer@hostname".

You should remove (unlink) the PPD file after you are done using it. The filename is stored in a static buffer and will be overwritten with each call to `cupsGetPPD()`.

Example

```
#include <cups/cups.h>

char *ppd;

...

ppd = cupsGetPPD("printer@hostname");

...

unlink(ppd);
```

cupsGetPrinters()

Usage

```
int
cupsGetPrinters(char ***printers);
```

Arguments

Argument	Description
printers	Pointer to character pointer array.

Returns

The number of printer printers available.

Description

`cupsGetPrinters()` gets a list of the available printers. The returned array should be freed using the `free()` when it is no longer needed.

Example

```
#include <cups/cups.h>

int  i;
int  num_printers;
char **printers;

...

num_printers = cupsGetPrinters(;

...

if (num_printers > 0)
{
  for (i = 0; i num_printers; i ++)
    free(printers[i]);

  free(printers);
}
```

See Also

cupsGetClasses(), cupsGetDefault()

cupsLangDefault()

Usage

```
const char *
cupsLangDefault(void);
```

Returns

A pointer to the default language structure.

Description

`cupsLangDefault()` returns a language structure for the default language. The default language is defined by the `LANG` environment variable. If the specified language cannot be located then the POSIX (English) locale is used.

Call `cupsLangFree()` to free any memory associated with the language structure when you are done.

Example

```
#include <cups/language.h>

cups_lang_t *language;
...

language = cupsLangDefault();

...

cupsLangFree(language);
```

See Also

cupsLangEncoding(), cupsLangFlush(), cupsLangFree(), cupsLangGet(), cupsLangString()

cupsLangEncoding()

Usage

```
char *
cupsLangEncoding(cups_lang_t *language);
```

Arguments

Argument	Description
language	The language structure.

Returns

A pointer to the encoding string.

Description

`cupsLangEncoding()` returns the language encoding used for the specified language, e.g. "iso−8859−1", "utf−8", etc.

Example

```
#include <cups/language.h>

cups_lang_t *language;
char        *encoding;
...

language = cupsLangDefault();
encoding = cupsLangEncoding(language);
...

cupsLangFree(language);
```

See Also

cupsLangDefault(), cupsLangFlush(), cupsLangFree(), cupsLangGet(), cupsLangString()

cupsLangFlush()

Usage

```
void
cupsLangFlush(void);
```

Description

`cupsLangFlush()` frees all language structures that have been allocated.

Example

```
#include <cups/language.h>

...

cupsLangFlush();
```

See Also

cupsLangDefault(), cupsLangEncoding(), cupsLangFree(), cupsLangGet(), cupsLangString()

cupsLangFree()

Usage

```
void
cupsLangFree(cups_lang_t *language);
```

Arguments

Argument	Description
language	The language structure to free.

Description

`cupsLangFree()` frees the specified language structure.

Example

```
#include <cups/language.h>

cups_lang_t *language;
...

cupsLangFree(language);
```

See Also

cupsLangDefault(), cupsLangEncoding(), cupsLangFlush(), cupsLangGet(), cupsLangString()

cupsLangGet()

Usage

```
cups_lang_t *
cupsLangGet(const char *name);
```

Arguments

Argument	Description
name	The name of the locale.

Returns

A pointer to a language structure.

Description

`cupsLangGet()` returns a language structure for the specified locale. If the locale is not defined then the POSIX (English) locale is substituted.

Example

```
#include <cups/language.h>

cups_lang_t *language;

...

language = cupsLangGet("fr");

...

cupsLangFree(language);
```

See Also

cupsLangDefault(), cupsLangEncoding(), cupsLangFlush(), cupsLangFree(), cupsLangString()

cupsLangString()

Usage

```
char *
cupsLangString(cups_lang_t *language,
               int         message);
```

Arguments

Argument	Description
language	The language to query.
message	The message number.

Returns

A pointer to the message string or `NULL` if the message is not defined.

Description

`cupsLangString()` returns a pointer to the specified message string in the specified language.

Example

```
#include <cups/language.h>

cups_lang_t *language;
char        *s;
...

language = cupsLangGet("fr");

s = cupsLangString(language, CUPS_MSG_YES);

...

cupsLangFree(language);
```

See Also

cupsLangDefault(), cupsLangEncoding(), cupsLangFlush(), cupsLangFree(), cupsLangGet()

cupsLastError()

Usage

```
ipp_status_t
cupsLastError(void);
```

Returns

An enumeration containing the last IPP error.

Description

`cupsLastError()` returns the last IPP error that occurred. If no error occurred then it will return `IPP_OK` or `IPP_OK_CONFLICT`.

Example

```
#include <cups/cups.h>

ipp_status_t status;

...

status = cupsLastError();
```

See Also

cupsCancelJob(), cupsPrintFile()

cupsMarkOptions()

Usage

```
int
cupsMarkOptions(ppd_file_t *ppd,
                int num_options,
                cups_option_t *options);
```

Arguments

Argument	Description
ppd	The PPD file to mark.
num_options	The number of options in the options array.
options	A pointer to the options array.

Returns

The number of conflicts found.

Description

`cupsMarkOptions()` marks options in the PPD file. It also handles mapping of IPP option names and values to PPD option names.

Example

```
#include <cups/cups.h>

int           num_options;
cups_option_t *options;
ppd_file_t    *ppd;

...

cupsMarkOptions(ppd, num_options, options);
```

See Also

cupsAddOption(), cupsFreeOptions(), cupsGetOption(), cupsParseOptions()

cupsParseOptions()

Usage

```
int
cupsParseOptions(const char *arg,
                 int num_options,
                 cups_option_t **options);
```

Arguments

Argument	Description
arg	The string containing one or more options.
num_options	The number of options in the options array.
options	A pointer to the options array pointer.

Returns

The new number of options in the array.

Description

`cupsParseOptions()` parses the specifies string for one or more options of the form "name=value", "name", or "noname". It can be called multiple times to combine the options from several strings.

Example

```
#include <cups/cups.h>

int           num_options;
cups_option_t *options;

...

num_options = 0;
options     = (cups_option_t *)0;
num_options = cupsParseOptions(argv[5], num_options, &options);
```

See Also

cupsAddOption(), cupsFreeOptions(), cupsGetOption(), cupsMarkOptions()

cupsPrintFile()

Usage

```
int
cupsPrintFile(const char *printer,
              const char *filename,
              const char *title,
              int num_options,
              cups_option_t *options);
```

Arguments

Argument	Description
printer	The printer or class to print to.
filename	The file to print.
title	The job title.
num_options	The number of options in the options array.
options	A pointer to the options array.

Returns

The new job ID number or 0 on error.

Description

`cupsPrintFile()` sends a file to the specified printer or class for printing. If the job cannot be printed the error code can be found by calling `cupsLastError()`.

Example

```
#include <cups/cups.h>

int           num_options;
cups_option_t *options;

...

cupsPrintFile("printer@hostname", "filename.ps", "Job Title", num_options,
              options);
```

See Also

cupsCancelJob(), cupsLastError()

cupsRasterClose()

Usage

```
void
cupsRasterClose(cups_raster_t *ras);
```

Arguments

Argument	Description
ras	The raster stream to close.

Description

`cupsRasterClose()` closes the specified raster stream.

Example

```
#include <cups/raster.h>

cups_raster_t *ras;

...

cupsRasterClose(ras);
```

See Also

cupsRasterOpen(), cupsRasterReadHeader(), cupsRasterReadPixels(), cupsRasterWriteHeader(), cupsRasterWritePixels()

cupsRasterOpen()

Usage

```
cups_raster_t *
cupsRasterOpen(int fd,
               cups_mode_t mode);
```

Arguments

Argument	Description
fd	The file descriptor to use.
mode	The mode to use; CUPS_RASTER_READ or CUPS_RASTER_WRITE.

Returns

A pointer to a raster stream or NULL if there was an error.

Description

cupsRasterOpen() opens a raster stream for reading or writing.

Example

```
#include <cups/raster.h>

cups_raster_t *ras;

...

ras = cupsRasterOpen(0, CUPS_RASTER_READ);
```

See Also

cupsRasterClose(), cupsRasterReadHeader(), cupsRasterReadPixels(), cupsRasterWriteHeader(), cupsRasterWritePixels()

cupsRasterReadHeader()

Usage

```
unsigned
cupsRasterReadHeader(cups_raster_t *ras,
                     cups_page_header_t *header);
```

Arguments

Argument	Description
ras	The raster stream to read from.
header	A pointer to a page header structure to read into.

Returns

1 on success, 0 on EOF or error.

Description

`cupsRasterReadHeader()` reads a page header from the specified raster stream.

Example

```
#include <cups/raster.h>

int                  line;
cups_raster_t        *ras;
cups_raster_header_t header;
unsigned char        pixels[8192];
...

while (cupsRasterReadHeader(ras, &header))
{
  ...

  for (line = 0; line < header.cupsHeight; line ++)
  {
    cupsRasterReadPixels(ras, pixels, header.cupsBytesPerLine);

    ...
  }
}
```

See Also

cupsRasterClose(), cupsRasterOpen(), cupsRasterReadPixels(), cupsRasterWriteHeader(), cupsRasterWritePixels()

cupsRasterReadPixels()

Usage

```
unsigned
cupsRasterReadPixels(cups_raster_t *ras,
                     unsigned char *pixels,
                     unsigned length);
```

Arguments

Argument	Description
ras	The raster stream to read from.
pixels	The pointer to a pixel buffer.
length	The number of bytes of pixel data to read.

Returns

The number of bytes read or 0 on EOF or error.

Description

`cupsRasterReadPixels()` reads pixel data from the specified raster stream.

Example

```
#include <cups/raster.h>

int                  line;
cups_raster_t        *ras;
cups_raster_header_t header;
unsigned char        pixels[8192];
...

while (cupsRasterReadHeader(ras, &header))
{
  ...

  for (line = 0; line < header.cupsHeight; line ++)
  {
    cupsRasterReadPixels(ras, pixels, header.cupsBytesPerLine);

    ...
  }
}
```

See Also

cupsRasterClose(), cupsRasterOpen(), cupsRasterReadHeader(), cupsRasterWriteHeader(), cupsRasterWritePixels()

cupsRasterWriteHeader()

Usage

```
unsigned
cupsRasterWriteHeader(cups_raster_t *ras,
                      cups_page_header_t *header);
```

Arguments

Argument	Description
ras	The raster stream to write to.
header	A pointer to the page header to write.

Returns

1 on success, 0 on error.

Description

`cupsRasterWriteHeader()` writes the specified page header to a raster stream.

Example

```
#include <cups/raster.h>

int                  line;
cups_raster_t        *ras;
cups_raster_header_t header;
unsigned char        pixels[8192];
...

cupsRasterWriteHeader(ras, &header);

for (line = 0; line < header.cupsHeight; line ++)
{
  ...

  cupsRasterWritePixels(ras, pixels, header.cupsBytesPerLine);
}
```

See Also

cupsRasterClose(), cupsRasterOpen(), cupsRasterReadHeader(), cupsRasterReadPixels(), cupsRasterWritePixels()

cupsRasterWritePixels()

Usage

```
unsigned
cupsRasterWritePixels(cups_raster_t *ras,
                      unsigned char *pixels,
                      unsigned length);
```

Arguments

Argument	Description
ras	The raster stream to write to.
pixels	The pixel data to write.
length	The number of bytes to write.

Returns

The number of bytes written.

Description

`cupsRasterWritePixels()` writes the specified pixel data to a raster stream.

Example

```
#include <cups/raster.h>

int                  line;
cups_raster_t        *ras;
cups_raster_header_t header;
unsigned char        pixels[8192];
...

cupsRasterWriteHeader(ras, &header);

for (line = 0; line < header.cupsHeight; line ++)
{
  ...

  cupsRasterWritePixels(ras, pixels, header.cupsBytesPerLine);
}
```

See Also

cupsRasterClose(), cupsRasterOpen(), cupsRasterReadHeader(), cupsRasterReadPixels(), cupsRasterWriteHeader()

cupsServer()

Usage

```
const char *
cupsServer(void);
```

Returns

A pointer to the default server name.

Description

`cupsServer()` returns a pointer to the default server name. The server name is stored in a static location and will be overwritten with every call to `cupsServer()`

The default server is determined from the following locations:

1. The `CUPS_SERVER` environment variable,
2. The `ServerName` directive in the *cupsd.conf* file,
3. The default host, "localhost".

Example

```
#include <cups/cups.h>

const char *server;

server = cupsServer();
```

See Also

cupsGetPassword(), cupsUser()

cupsTempFile()

Usage

```
char *
cupsTempFile(char *filename,
             int length);
```

Arguments

Argument	Description
filename	The character string to hold the temporary filename.
length	The size of the filename string in bytes.

Returns

A pointer to `filename`.

Description

`cupsTempFile()` generates a temporary filename for the */var/tmp* directory or the directory specified by the `TMPDIR` environment variable.

Example

```
#include <cups/cups.h>

char filename[256];

cupsTempFile(filename, sizeof(filename));
```

cupsUser()

Usage

```
const char *
cupsUser(void);
```

Returns

A pointer to the current username or `NULL` if the user ID is undefined.

Description

`cupsUser()` returns the name associated with the current user ID as reported by the `getuid()` system call.

Example

```
#include <cups/cups.h>

const char *user;

user = cupsUser();
```

See Also

cupsGetPassword(), cupsServer()

httpBlocking()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpCheck()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpClearFields()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpClose()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpConnect()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpDecode64()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpDelete()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpEncode64()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpError()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpFlush()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpGet()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpGets()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpGetDateString()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpGetDateTime()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpGetField()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpGetLength()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpHead()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpInitialize()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpOptions()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpPost()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpPrintf()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpPut()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpRead()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpReconnect()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpSeparate()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpSetField()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpTrace()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpUpdate()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

httpWrite()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddBoolean()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddBooleans()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddDate()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddInteger()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddIntegers()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddRange()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddRanges()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddResolution()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddResolutions()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddSeparator()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddString()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippAddStrings()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippDateToTime()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippDelete()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippFindAttribute()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippLength()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippNew()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippPort()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippRead()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippTimeToDate()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ippWrite()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeAddFilter()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeAddType()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeAddTypeRule()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeDelete()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeFileType()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeFilter()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeLoad()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeMerge()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeNew()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

mimeType()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdClose()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdConflicts()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

pddEmitFd()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdEmit()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdFindChoice()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdFindMarkedChoice()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdFindOption()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdIsMarked()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdMarkDefaults()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdMarkOption()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdOpenFd()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdOpenFile()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdOpen()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdPageLength()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdPageSize()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also

ppdPageWidth()

Usage

Arguments

Argument	Description

Returns

Description

Example

See Also